Integrating Computational Thinking in STEM Education: A Literature Review

(accepted on Sep. 27th 2021 by *International Journal of Mathematics and Science Education,* this is a preprint version.)

Changzhao Wang[1], Ji Shen[1], Jie Chao[2]

[1] Department of Teaching and Learning, University of Miami, Coral Gables, FL, USA

[2] Concord Consortium, Concord, MA, USA

Author Note

Changzhao Wang https://orcid.org/0000-0002-4939-347X

Ji Shen https://orcid.org/0000-0003-3267-3261

Jie Chao https://orcid.org/0000-0002-9153-3552

We have no known conflict of interest to disclose.

Correspondence concerning this article should be addressed to Ji Shen, Department of Teaching and Learning, University of Miami, 5202 University Drive, Coral Gables, FL 33124; Phone: 305-284-4970; Fax: (305) 284-6998; Email: ji.shen1221@gmail.com

Abstract

Research focusing on the integration of computational thinking (CT) into science, technology, engineering, and mathematics (STEM) education started to emerge. We conducted a semi-systematic literature review on 55 empirical studies on this topic. Our findings include: (a) the majority of the studies adopted domain-general definitions of CT and a few proposed domain-specific CT definitions in STEM education; (b) the most popular instructional model was problem-based instruction and the most popular topic contexts included game design, robotics, and computational modelling; (c) while the assessments of student learning in integrated CT and STEM education targeted different objectives with different formats, about a third of them assessed *integrated* CT and STEM; (d) about a quarter of the studies reported differential learning processes and outcomes between groups, but very few of them investigated how pedagogical design could improve equity. Based on the findings, suggestions for future research and practice in this field are discussed in terms of operationalizing and assessing CT in STEM contexts, instructional strategies for integrating CT in STEM, and research for broadening participation in integrated CT and STEM education.

Keywords: Computational Thinking, STEM Education, Instructional Strategies, Assessment, Equity

Introduction

Computational thinking (CT) is considered necessary for everyone and everyday life in the new century (Wing, 2006, 2008). It is the practice or thought process of applying fundamental computer science concepts to solve problems (Aho, 2012; Wing, 2006, 2008). More specifically, it includes extracting key information from the concrete details of a problem (abstraction), reformulating a larger problem into a set of smaller ones (decomposition), detecting the patterns embedded in data, developing and applying algorithms, and so forth (College Board, 2020; Wing, 2006, 2008).

In recent years, there is an emerging trend of integrating computing (hence CT) into disciplinary education, especially in the Science, Technology, Engineering, and Mathematics (STEM) fields (Lee et al., 2020; Li et al., 2020). Research has shown that the STEM contents and contexts can benefit CT learning (Weintrop et al., 2014; Orton et al., 2016). Likewise, incorporating CT into STEM education will also enhance students' learning of STEM contents (Repenning, Webb, & Ioannidou, 2010; Sengupta, Kinnebrew, Basu, Biswas, & Clark, 2013; Wilensky & Reisman, 2006; Lewis & Shah, 2012), because of the integral role of computation in modern STEM disciplines (Martin, 2018). For instance, the Next Generation Science Standards (NGSS) framework (National Research Council [NRC], 2012) lists "using mathematics and computational thinking" as one of the eight core practices in K–12 science classrooms.

More importantly, integrating CT into STEM education has the potential of reducing inequity in terms of CT learning (Weintrop et al., 2014). Traditional CT education (i.e., when not embedded in STEM education) is often based on computers and programming courses (Grover & Pea, 2013). However, computer science or programming courses are still limited,

especially in lower grades in K–12 settings; even if the courses are accessible to all students, many will not take them because of lacking interest or performing badly in programming, since computer science courses are usually not mandatory (Code.org, CSTA, & ECEP Alliance, 2020). More students would have the opportunity to learn CT when it is taught in STEM classrooms, as STEM courses are more widely offered and are more likely to be compulsory than computer science courses (Weintrop et al., 2014).

With the ongoing discussions about what CT means, there has been little consensus on how it should be operationalized in education (Barr & Stephenson, 2011; Grover & Pea, 2013; Román-González, Pérez-González, & Jiménez-Fernández, 2017). For example, the International Society for Technology in Education (ISTE) and Computer Science Teachers Association (CSTA) in 2011 defined CT in K–12 education as a problem-solving process that includes formulating problems, using a computer or other tools, logically organizing, analyzing, and representing data, automating solutions through algorithmic thinking, achieving efficient and effective solutions, and generalizing and transferring to other problems. Brennan and Resnick (2012) developed a CT framework specifically for learning in the programming environment. Thus, they identified seven programming concepts—sequences, loops, parallelism, events, conditionals, operators, and data—as CT concepts.

Crosscutting with other disciplinary fields further contributes to the wide diversity of CT definitions. For example, Weintrop et al. (2016) framed CT in math and science education as four categories of practices, including data practices, modeling and simulation practices, computational problem-solving practices, and systems thinking practices. This is in accordance with the rise of the focus on practices in STEM education (NRC, 2012). More recently, based on their analyses of the activities collected from researchers and practitioners that integrated CT into K–12 STEM classrooms, Lee and Malyn-Smith (2020) defined CT

from a disciplinary perspective that included five CT integration elements—understanding complex systems, innovating with computational representations, designing solutions that leverage computational power and resources, engaging in collective sense-making around data, and understanding potential consequences of actions.

Besides its diverse theoretical interpretations, integrating CT in STEM education remains challenging as many practical issues are still under exploration, such as what activities and approaches are effective for integrating CT in various STEM contexts, and how CT should be assessed in the new context (Grover & Pea, 2013; Li et al., 2020; Shute, Sun, & Asbell-Clarke, 2017; Tang et al., 2020). These aspects need to be sorted out so that educators could deliver effective integrated CT and STEM instruction.

For further contributing to the research and practice of this important field, it is necessary to probe the current state of how these issues have been addressed. Therefore, we conducted a literature review of empirical studies on this topic with the following guiding questions: (to address the diverse theoretical interpretations) (1) How is CT defined in STEM education? (to address the practical concerns of integrating CT in STEM education) (2) What instructional strategies are used to incorporate CT into STEM education? (3) How are CT and STEM contents assessed in integrated CT and STEM education? (to see if sufficient work has been done as one of the major reasons of integrating CT into STEM education is to improve equity) (4) What aspects of equity-related issues are investigated and reported in the literature on integrating CT in STEM?

Method

The current study is a semi-systematic literature review (Snyder, 2019) as we conducted content analysis on both qualitative and quantitative studies to identify general trends in the field (Çalık & Sözbilir, 2014). A semi-systematic approach is appropriate for topics that are “conceptualized differently and studied by various groups of researchers

within diverse disciplines and that hinder a full systematic review process" (Snyder, 2019, p. 335). Many relevant studies termed CT differently and focused a wide range of integration disciplines, making an exhaustive literature search impossible. Yet, we followed a rigorous process (see below) to collect and analyze the literature.

Literature Search and Selection

We first determined a set of inclusion/exclusion criteria: (a) The paper to be included covers both CT and STEM education and stresses integrating CT in STEM subjects. STEM refers to one or more (sub)disciplinary contexts—being isolated (e.g., physics) or integrated (e.g., biomechanical engineering) in nature—within the broadly defined STEM field where CT is applied. (b) The paper to be included reports at least one empirical research study that engages participants in CT activities. We excluded studies that mainly focused on describing a curriculum that integrated CT with STEM but did not report research or assessment data on participants' learning. (c) We only included literature published in English. (d) We excluded works published in book chapters or conference proceedings of the same study that was published in a journal.

We collected literature through three means: searching databases and popular search engines, examining citations in collected articles, and including articles to the knowledge of the authors. In terms of the database approach, we confined the concept of CT to be explicitly expressed as "computational thinking." First, we searched Educational Research Information Center (ERIC) and ACM Digital Library for articles that included "computational thinking," "education," and any of the following five terms— "STEM," "science," "technology," "engineering," and "mathematics"—in any field of the article. The returned results did not provide enough empirical studies. Then we adjusted keywords, and added discipline-specific terms such as "physics," "chemistry," and "biology" to supplement "science." We also added "game design" and "robotics" to supplement "technology" because they emerged to be two

main technological contexts for CT based on our observation of related articles. Given their popularity, Google Scholar and ResearchGate were used as supplementary search sources.

We conducted a total of two rounds of literature search and selection. During the first round of search (September 2018 – January 2019), we obtained 87 articles. We also examined the citations from the literature we gathered and obtained another 20 articles. Having the initial pool of 107 articles, we applied the specified eligibility criteria on each article (by reading the whole set of articles) and 40 papers remained after this first round of selection. During the second round of search (January 2020 – early February 2020), the same keywords were used to search the same data sources, but with an additional constraint on the publication year (i.e., after 2018) to gather more recent publications. We also examined the articles from a very recent JOST special issue on CT (the papers of which were published online between November 2019 and February 2020). The second round resulted in 23 new papers for further selection, 15 of which were included into our library. In total, we collected 55 articles for this review (see Electronic Supplemental Material 1).

Analysis

The articles were reviewed one by one. First, the general information of each study was documented, including the issued/publication year, subject matter/discipline, grade level of the participants, where the study took place, etc. Then, specific notes were taken, and open codes were developed to address our research questions related to CT definition, instructional strategies, assessment, and equity related issues. The detailed coding book is in Electronic Supplemental Material 2 (the results can be found in Electronic Supplementary Material 1).

Regarding CT definition, we took notes about the definition(s) of CT that were listed in the article, and the definition that was used as framework for instruction, assessment, and/or data analysis. Additionally, to get an overall idea of how CT is generally defined or interpreted in these papers, we collected the theoretical section that is relevant with CT and

used the collection of texts to create a word cloud using an online word cloud generator WordClouds (https://www.wordclouds.com/). The tool generates both a list of unique words with the number of times they appear in the text sample and a visualization of words (with the sizes of which proportional to their frequencies). We deleted irrelevant words (e.g., "can," "define") with high frequencies, and transformed the morphological variants of words into the same form (e.g., replacing the plural form of a noun by the singular form).

When coding the instructional strategies for integrating CT into STEM education, we coded the instructional models (e.g., problem-based learning), the topic contexts (e.g., game design, science [biology]), ways of scaffolding (e.g., fading), types of CT related activities (e.g., programming or unplugged activities), and specific programming language or environment used (e.g., Scratch). We also took notes on whether or not explicit instruction on CT was incorporated, and if so, how.

In terms of assessments, we coded their format (e.g., survey, test, interview) and domain target (e.g., CT knowledge, attitudes towards CT, specific STEM content knowledge). If a study contained more than one assessment, each assessment was coded. If a study only mentioned that they administered an assessment but did not provide specific information, this assessment was not coded. When a study reported differentiated results among different student groups and populations, we took specific notes on if the study investigated equity-related issues such as gender difference in both their instructional strategies and assessments.

Coding Reliability

Since the reviewed literature is open-ended data, our coding chose the percent agreement as reliability index (Syed & Nelson, 2015). The coding reliability was established through rounds of discussion between the coders (the first two authors). We initially read and coded a set of articles individually to build the first version of the coding book. We carefully

discussed our results and drafted an operationalizable coding scheme. Then we independently coded a set of three articles (5%) and compared our coding results, which showed 92.1% agreement. All coding inconsistencies and disagreements were discussed and resolved. Then, the coding scheme was further clarified and modified accordingly. We repeated this process for another two rounds (three articles for each round) and had 93.7% and 96.8% before-discussion agreement, respectively. All inconsistencies and disagreement were resolved through discussion. Based on the finalized coding scheme, the first author finished coding the remaining articles. The codes for all 55 articles are in Electronic Supplemental Material 1.

Results

Basic Descriptive Statistics

In terms of geographic locations, the majority of the articles came from the USA (n=40). Other countries or regions included Canada (5), New Zealand (2), Croatia (1), England (1), Greece (1), Israel (1), Spain (1), Switzerland (1), Taiwan (1), and Turkey (1). In terms of growth over the years, there is clearly a rising trend: 2009 – 2011 (3), 2012 – 2014 (5), 2015 – 2017 (22), 2018 – 2020 (25; note that our search ended early 2020). In terms of subject matter (Figure 1), 19 papers were on integrating CT in science education; 17 in technology and/or engineering education; 9 in mathematics education; and 10 in some combination of STEM disciplines.

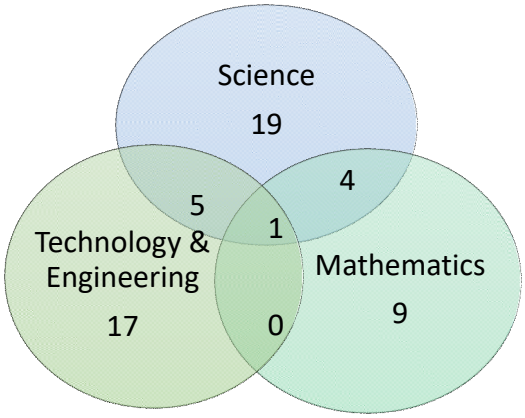

Figure 1. Number of papers for each one or combination of the disciplines.

In terms of participants, the middle school level had the most studies (n=18), followed by upper elementary level (n=11)[1] and high school level (n=9). There were 11 studies targeting teachers (5 for pre-service teachers and 6 for in-service teachers). Higher education had 3 studies and lower elementary had 4 studies.

Defining CT in STEM

Word cloud. Word cloud provides a quick overview of the frequently used vocabularies to define or interpret CT in the collection of literature. WordClouds produced one as shown in Figure 2. The words could be roughly divided into three levels based on the range of frequencies and the meaning of words.

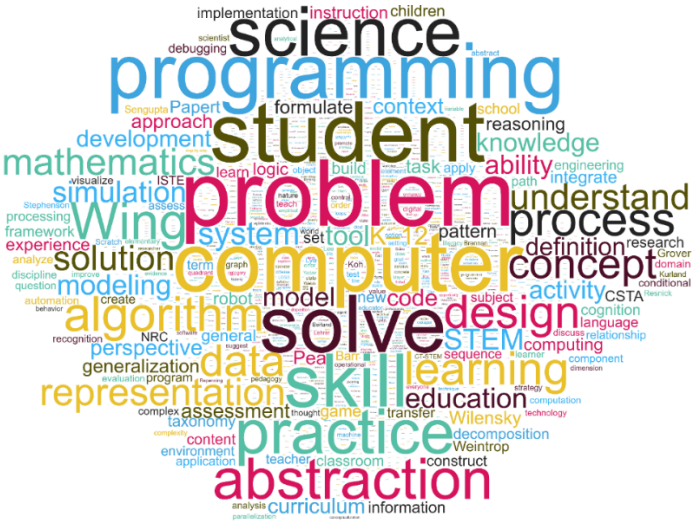

Figure 2. Word cloud of CT theoretical parts from all papers (The sizes of words are proportional to their counts, and the colours were randomly assigned by the tool).

The most prominent words at the first glance are the following: “problem” (f=170; times appeared in the text sample), “computer” (f=107), “student” (f=107), “solve” (f=102), “programming” (f=98), and “skill” (f=95). They represent the most fundamental concepts related with CT. *Problem-solving skill* is considered by many as the driving objective for applying or developing CT in STEM. All of the reviewed papers are in educational contexts,

[1] Djambong & Freiman (2016) included participants from both upper elementary and middle school levels, and it was counted in both categories.

therefore, the primary target is *student*. *Programming* has been used as the main form of activity in learning about CT in STEM and *computers* are the the primary tool.

Then we may find words that are displayed slightly smaller because of their lower frequencies. Yet, they are more specific in terms of the context and meaning of CT, such as, "science" (f=91), "practice" (f=85), "abstraction" (f=77), "Wing" (f=66), "algorithm" (f=61), "process" (f=61), "design" (f=59), "data" (f=56), "concept" (f=54), and "representation" (f=49). *Science* is one of the most common domain contexts for integrating CT. *Abstraction*, using *algorithm*, *processing data*, and *data representation* are some of the most common CT *concepts* and *practice*s (Chen, Shen, Barth-Cohen, Jiang, Huang, Eltoukhy, 2017), two of the three key dimensions of CT framework brought up by Brennan and Resnick (2012). System *design* (in engineering and technology) and algorithm design are core components of CT in the conceptualizations by ISTE and CSTA (2011) and *Wing* (2006).

The words that appeared below 49 times constitute the third level. They cover some additional *context*s and *practices* for integrating CT in *STEM*, such as "mathematics" (f=47), "system" (f=41), "simulation" (f=38), "STEM" (f=37), "modelling" (f=36), "code" (f=30), "formulate" (f=24), "generalization" (f=24), "pattern" (f=24), "robot" (f=22), "game" (f=21), "decomposition" (f=19), "engineering" (f=17), "debugging" (f=16).

Operationalizing CT in STEM education. The word cloud provides an overview but is not sufficient to address research question one. Here we further elaborate on our observation about how CT was operationalized in these studies (Some key frameworks of CT definition or operationalization are summarized in Table 1). These definitions or operationalization of CT can be categorized into two major types: generic ones (i.e., not attending to specific disciplines) and specific ones (i.e., attending to specific disciplines).

Table 1. *Key Frameworks of CT Definition or Operationalization in the Reviewed Articles*

Authorship	**Wing, 2006**	**ISTE & CSTA, 2011**	**Brennan & Resnick, 2012**	**Weintrop et al., 2016**

Description	CT involves the application of foundational computer science concepts or skills in solving problems, designing systems, and understanding human behaviour. The concepts/skills include reformulating problems, reduction, transformation, abstraction, decomposition, etc.	CT involves two dimensions: skills (e.g., formulating problems computationally, processing data logically, representing data through abstractions, automating solutions through algorithmic thinking) and dispositions (e.g., confidence, persistence, tolerance, the ability of dealing with open ended problems, the ability of collaboration).	CT involves three key dimensions: computational concepts (e.g., sequences, loops, conditionals, and data), computational practices (e.g., being iterative, debugging, remixing, and abstracting and modularizing) , and computational perspectives (e.g., understandings of selves, relationships to others, and the technological world around them).	CT involves four categories of practices integral in math and science education: data practices, modelling and simulation practices, computational problem-solving practices, and systems thinking practices.
STEM specificity	No	No	No	Yes

For the generic type, Wing's (2006, 2008) broad conceptualization was adopted in 10 papers and referenced by most of our articles. She made CT a popular term and proposed it should be treated at the same level as the three Rs: arithmetic, reading, and writing. ISTE and CSTA (2011) provided a more specific operational definition, which delineated concrete CT skills and CT dispositions in K – 12 education. They framed CT skills as a set of critical skills involved in a problem-solving process, such as formulating problems computationally, data processing for solving problems, automating solutions, and generalizing and transferring the problem-solving process. Closely related to CT skills, CT dispositions were also defined as significant components of CT, including confidence in dealing with complexity, persistence in working with difficult problems, tolerance for ambiguity, the ability to deal with open ended problems, and the ability to communicate and work with others to achieve a common goal or solution. This framework was referenced in seven articles in our literature: Two of them directly adopted its definition of CT skills for assessing students' learning of CT in the contexts of robotics and game design (i.e., Leonard, Buss, Gamboa, et al., 2016;

Leonard, Buss, Unertl et al., 2016); one of them directly used its definition of CT dispositions (i.e., Hadad et al., 2020); and the others adapted it to make their own CT definitions. Brennan and Resnick (2012) further expanded the interpretation of CT and proposed three key dimensions of CT: foundational CS concepts such as sequences and loops; computational practices such as testing and debugging; and computational perspectives that a learner applies in understanding selves, relationships to others, and the technological world around them. This framework was adopted in two studies (i.e., Boticki, Pivalica, & Seow, 2018; Falloon, 2016) for the learning of students aged 5 – 8 years old in a block-based programming environment (e.g., ScrachJr).

Comparing to the generic CT definitions or interpretations, many articles also described specific perspectives of CT pertaining to the STEM fields. Weintrop et al. (2016) proposed a taxonomy of four categories of CT practices (i.e., data, modelling and simulation, computational problem solving, and systems thinking), which was adopted by nine articles. The practices were extracted from the commonalities shared by well-acknowledged CT practices and common math and science practices. For example, modelling and simulation are essential practices for scientists and mathematicians (NGSS Lead States, 2013), and they are also considered as essential CT skills (ISTE & CSTA, 2011; Wing, 2006). The framework was adopted by nine articles, eight of which directly used it as the operational definition for their curriculum and/or assessment.

The majority of the remaining studies developed their own operational CT framework by integrating some extant definitions. Based on our reading, the studies that shared similar learning environments or technological platforms tended to identify the same or similar CT components or constructs as their operational CT framework. For example, the CT frameworks in the only three studies using maker activities overlapped with each other to a great extant: Campbell and Heller (2019) framed CT constructs as decomposition,

abstraction, pattern recognition, and algorithmic thinking; Hadad et al. (2020) included decomposition, abstraction, pattern recognition, algorithm design, and evaluation; and Yin et al. (2020) broke down CT to decomposition, abstraction, algorithmic thinking, and pattern generalization.

Instructional Strategies for Integrating CT in STEM

Accompanied with the diverse definitions, a variety of instructional strategies have been used or explored. Some of them are popular among the reviewed studies, and they are summarized as instructional models and topic contexts, programming and non-programming activities, and ways of scaffolding student learning.

Instructional models and topic contexts. A total of 11 studies explicitly used the instructional model of Problem-Based or problem-driven Instruction (PBI). These PBI strategies centred around certain problems designed by the researchers so that students practiced and improved CT or content knowledge and skills while working on those problems. Typical problems included programming problems (e.g., programming game characters to sort out flags of different colours, Witherspoon, Higashi, Schunn, Baehr, & Shoop, 2017), mathematics problems (e.g., calculating areas of polygons, Pei, Weintrop, & Wilensky, 2018), and science problems (e.g., delivering medicines to a tribe in a remote area, Hutchins et al., 2020).

Our analysis also shows that, compatible with PBI, the most popular topic contexts are game design, robotics, and computational modelling. Game design represents the most common topic context for cultivating students' CT skills in STEM in our pool (12 studies). It has the advantage of quickly drawing students' attention (Bremner, 2013), and providing students with opportunities to practice programming and to learn STEM content at the same time (Hoover et al., 2016; Leonard, Buss, Gamboa, et al., 2016; Wu, 2018). Among these studies, the most popular programming environment for game design was Scratch (a block-

based programming environment). For example, in the study reported by Hoover et al. (2016), students were asked to create games about climate change using Scratch. The games created by students reflected rather deep understanding of climate change because they not only covered scientific concepts (e.g., greenhouse gases, photosynthesis), but also demonstrated the mechanisms of global warming and included measures to prevent it.

Robotics education becomes increasingly popular because robots can easily motivate and engage students in both engineering and technology practices (Bers, Flannery, Kazakoff, & Sullivan, 2014; Leonard, Buss, Gamboa, et al., 2016). Among the 12 studies using the robotics context, one study used a humanoid robot (Chen et al., 2017) and the rest all used non-humanoid robotic kits (e.g., LEGO EV3) that required participants to build their own robots from bricks, motors, and sensors first and then program them to accomplish tasks or solve problems. For example, Bers et al. (2014) reported that their construction-based robotics curriculum toward kindergarteners consisted of an introduction lesson about general engineering design, the second lesson to learn to make robots by themselves, then lessons on how to program the robots, and that learners showed many positive results on robotics and CT (e.g., significant improvement in correspondence, sequencing, and debugging skills). Berland and Wilensky (2015) conducted a study comparing student learning with a virtual robot and that with a physical robot and found that the two different forms led to students' differentiated perspectives to interpret situations, but similar outcome of students' CT skills, as measured by tests on students' capability to follow and change a flowchart.

In addition to robotics and game design, computational modelling was adopted as topic context in 16 studies. Scientific modelling is the process of creating representations, rules, or reasoning structures for predictions and explanations of science phenomena or mechanisms (Schwarz & White, 2005; Namdar & Shen, 2015). Computational modelling refers to the use of computational tools to carry out modelling tasks. Aksit and Wiebe (2020)

reported a five-day intervention embedded in a seventh-grade science classroom; during the intervention, students were asked to build two models with Scratch: one that would simulate the motion of a car on a frictionless path and the other that would simulate the free fall of a basketball. In another study (Bortz, Gautam, Tatar, & Lipscomb, 2020), middle school students carried out computational chemistry tasks using the NetLogo modelling environment. An initial model (e.g., the model of the forming and splitting of water molecules) was provided before students worked in groups to evaluate it and make changes on the code if necessary.

Programming and non-programming activities. Programming is a common vehicle to help cultivate students' CT since programming involves creating instructions through a set of computer readable representations (i.e., programming language) to make computers to accomplish certain tasks or solve problems.

Specific consideration is often given to leverage learners' age appropriateness and/or prior experience with programming. Text-based programming language was used in 7 studies that involved students in middle school (n=1), high school (n=1), or higher education (n=2), and pre-/in- service teachers (n=3). For example, R was used to involve high school students in mathematical modelling (Wiedemann, Chao, Galluzzo, Simoneau, 2020) and Python programming tasks were designed to introduce CT to undergraduates who majored in science (Hambrusch et al., 2009). Nevertheless, the majority of our papers (n=38) described approaches using block-based programming languages, such as Scratch and Lego Mindstorm (a block-based programming environment for Lego robotics). These block-based programming environments are intuitive to use and are designed mainly for young learners or novice programmers. For instance, Benton, Hoyles, Kalas, & Noss (2017) reported a ScratchMaths project in which curriculums were designed to integrate Scratch programming and mathematics for fifth graders. In one of their activities, students were asked to program

the Earth to be surrounded by objects one by one, in the way that petals surround the centre of a sunflower. To achieve this goal, they needed to fully understand what a 360° total turn means mathematically and create proper Scratch algorithm.

In spite of the popularity of computer programming, not all studies resorted to computers in their instructions. Embodied or unplugged activities were used for younger children or discussed by pre-/in- service teachers in seven studies. These activities typically involved physical body movements or used certain objects to represent or illustrate abstract concepts or principles (Barth-Cohen, Montoya, & Shen, 2019). For instance, Sung, Ahn, and Black (2017) investigated the effects of the level of embodiment on kindergarten and first-grade students' mathematics and programming learning. The students in full embodiment groups were asked to complete the number lines on the floor by either physically performing the number or manipulating another student to perform, while students in low embodiment groups were asked to solve the problems directly on papers by themselves or by surrogating another. The post-assessments showed that students in full embodiment groups demonstrated significantly higher basic numeracy abilities than the other groups. To illustrate the concepts related to electrical circuits to seventh to ninth graders, Feldhausen, Weese, and Bean (2018) also took advantage of unplugged activities, including using marbles flowing in the hula-hoop to represent electrical current.

Scaffolding. Scaffolding refers to the process and aids provided to a learner to help him/her to learn a topic, solve a problem, or accomplish a goal (Devolder, van Braak, & Tondeur, 2012; Bakker, Smit, & Wegerif, 2015). Given that the studies we examined were conducted in the context of STEM education, it is important to see how instruction in these studies brought about CT. Explicit explanation of CT concepts in the instruction process was emphasized in 13 studies. Cateté et al. (2018) conducted a pilot study with in-service teachers to infuse CT into middle school life science classes. They showed that the teachers opposed

to the idea of implicitly integrating CT concepts, but proactively sought help from the research team to identify CT concepts hidden behind the activities in order to understand better. Feldhausen et al. (2018) also reported a study in which teachers illustrated clearly to fifth- and sixth-grade students how a programmer would think when they worked on the Scratch programming task.

A total of eight studies provided detailed information about the scaffolding strategy with respect to CT, and three of them conducted experiments to investigate the effectiveness of their scaffolding. In terms of fading (i.e., gradual removal of instructional support), Jaipal-Jamani and Angeli (2017) applied the scaffolded process starting from full scaffolding, then to partial scaffolding, finally to no scaffolding, and showed positive results in enhancing learners' CT skills. Lamprou and Repenning (2018) and Witherspoon et al. (2017) applied similar fading method and also showed positive results. In terms of sources of feedback, Sengupta et al. (2013) compared students who received one-on-one guidance with those who shared one classroom teacher in a learning environment that integrated CT and science learning, and showed that the one-on-one scaffolded group outperformed the comparison group. Likewise, Basu, Biswas, and Kinnebrew (2017) developed an adaptive scaffolding environment which supported students by recording their performance and then providing individualized scaffolds, and conducted a comparative study on sixth-grade students to investigate the effectiveness of this environment. The result showed that students who received adaptive scaffolding performed better than the other students in terms of modelling performance and behaviour, modelling strategy use, and learning of science and CT.

Assessments in Integrated CT and STEM Education

The targets of assessments. The targets of the assessments used in these studies can be categorized into three types: to assess CT related constructs only (n=18), to assess STEM related constructs only (n=1), and to assess both CT and STEM (n=36).

CT or CT related constructs included the affective domain and the cognitive domain. The former included learners' self-efficacy, attitudes, or interest in CT. For instance, Kalogiannakis and Papadakis (2017) conducted pre- and post- surveys on preservice teachers' self-efficacy in CT (Bean, Weese, Feldhausen, & Bell, 2015). The survey contained items like "I feel confident writing simple programs for the computer", and "I can identify how programming concepts relate to NGSS." Similar to this example, the instruments in most of our reviewed articles which claimed to assess affective domain towards CT, actually measured that towards programming in their items. In other words, they equated CT to programming in their assessment.

The cognitive domain included CT competency, CT understanding, and the integration of CT and STEM. In terms of assessing CT competency, likewise, many assessments depended on, or at least were highly related to programming. For instance, Orton et al. (2016) conducted a study aiming to incorporate CT into science and math education. In this study, apart from assessing students' attitudes toward STEM fields, they developed an instrument to assess CT skills. The pre- and post- assessments for CT skills were computer-based, and corresponded to the four-category taxonomy of CT in science and mathematics (Weintrop et al., 2016). In one sample question, a situation was depicted by both words and images and the question was to predict the result of a computer program written for the situation.

To assess CT understanding, Lamprou and Repenning (2018) asked students a question "What is CT for you?" in the middle of the course, and analysed their answers by categorizing them into seven major types, including thinking like a computer, programming/CS/computer work, problem division, thinking with the computer, problem solving, CT process (abstraction, automation, analysis), and other.

Among the 36 studies that assessed both CT and STEM, it is important to point out that only about half of the studies (n=20) assessed CT and STEM content in an integrated way. Five of them designed tests to assess the integration of CT and STEM content (i.e., Arastoopour Irgens et al., 2020; Bortz et al., 2020; Swanson, Anton, Bain, Horn, & Wilensky, 2019; Wiedemann et al., 2020; Yin, Hadad, Tang, & Lin, 2020). For instance, with the CT Integrated Achievement Test developed by the research team, Yin and colleagues (2020) assessed the integration of CT skills and physics and engineering content knowledge that was emphasized in the maker activities in their implementation (e.g., learn and build electric circuits, design and create e-textiles by sewing electric circuits on materials such as clothes or hats).

The remaining studies assessed the integrated CT and STEM by analysing students' artifacts (e.g., computer game products, Harrison et al., 2018), video records (e.g., classroom video, Farris et al., 2016), observation notes and/or reflections by teachers/students (e.g., Gadanidis, Clements, & Yiu, 2018), and students' log data collected by the CT tool (e.g., Botticki et al., 2018). In these data sources, students' performances on CT and STEM were integrated, so the assessments generally took the approach of applying a coding framework to identify the measures of CT and STEM components and then to further analyse them quantitatively and/or qualitatively. For example, Gadanidis, Cendros, Floyd, and Namukasa (2017) reported a study on integrating CT into mathematics teacher education. The participants' CT and math learning, as well as attitudes toward the integration of CT and math, were assessed through their reflection assignments and mind-map products. CT and math learning was framed as four themes (i.e., integration of CT and math, and Scratch for math learning, CT in other subjects, and CT in society). Attitudes toward the integration of CT and math fell into two categories: concern and anxiety, and a growth mindset.

Assessment formats. There were a variety of assessment formats used in the studies, including typical classroom assessment methods (e.g., tests, student artifacts) and research data collection methods (e.g., surveys, classroom observation, interviews, field notes, video records, activity logs).

Survey was the most popular form of assessment in these studies (n=26). The majority of the surveys focused on students' self-efficacy, attitude, experience, or interest toward CT or STEM contents. Some surveys also investigated participants' understanding of CT (e.g., the question of "what is CT for you"; Lamprou & Repenning, 2018) and their prior programming experience or knowledge (e.g., Feldhausen et al., 2018).

A total of 21 studies included knowledge and/or skill tests as their assessments. Among these, 2 focused on assessing STEM only, 10 focused on assessing CT only, and 9 on CT and STEM simultaneously (5 on integrated CT and STEM, and 4 assessed them independently).

Some studies adopted existing test instruments, including the Computational Thinking Test developed by Román-González et al. (2017), and the widely acclaimed tool Bebras (https://www.bebras.org/?q=examples). Since there are very limited number of CT tests available, most of the instruments used in these studies were developed by the researchers themselves and differ significantly. For example, Shen and co-authors (Chen et al., 2017; Shen et all, 2020) developed a CT assessment instrument based on the idea that CT is a broad construct applicable to many everyday contexts. The rubrics included five CT components: using established syntax, processing data, applying algorithm, representing solutions in multiple ways, and solving problems in an efficient manner.

A total of 15 studies used learners' artifacts as a major source of CT assessment, including robotic programs, learner-designed games, computer models, and written reflections. Different rubrics were developed and applied in these studies. For example, Bers

et al. (2014) assessed children's CT development by scoring the following four elements on their robotic programs: debugging (trouble-shooting), correspondence (choosing the right programming commands), sequencing (putting the commands in the right sequence), and control flow (realizing the same result by alternative ways of programming). As another example, to examine students' CT ability, Sung et al. (2017) assessed their ScratchJr programming products based on the number of errors they made and the number of skills they demonstrated.

Equity Issues when Integrating CT in STEM

Although many articles did report their participants' demographic information, only about a quarter of the total articles (n=14) investigated equity related issues in their studies.

Gender difference. Eleven studies analysed gender difference in various measures related to CT. With respect to CT competency, five studies found no statistically significant gender difference in CT competency among their participants (Duncan & Bell, 2015; Jenson & Droumeva, 2016; Mouza, Marzocchi, Pan, & Pollock, 2016; Orton et al., 2016; Witherspoon et al., 2017). These studies all involved a sample size greater than 40. Two other studies did show that female students performed better than male students, but with much smaller sample sizes: one was based on the observation of two iterations with less than 30 middle school students (less than 10 females) in each iteration (Grover, Pea, & Cooper, 2015); the other was based on the comparison of the CT scores of four focal students (two females and two males) (Leonard, Buss, Unertl et al., 2016).

In terms of affective domain related to CT, three studies found that females demonstrated lower interest, confidence, or self-efficacy than males. Among them, two studies drew the conclusion according to the *t*-test results of survey data collected from over 200 secondary school students (Duncan & Bell, 2015; Orton et al., 2016); one study reported that female pre-service elementary teachers were more confident in teaching science or

engineering under the context of integrated CT and STEM education than males, but without statistical analysis due to the small sample size (5 males and 26 females in total) (Campbell & Heller, 2019). Three other studies showed that, there was no statistically significant gender difference in the overall score of affective measurements, yet gender difference existed in some aspects of the measurements (Feldhausen et al., 2018; Jenson & Droumeva, 2016; Mouza et al., 2016). For example, Mouza et al. (2016) found that although boys and girls presented similar levels in many aspects (e.g., enjoying using computer, feeling motivated to succeed in computing, and intention to persist), boys scored statistically significantly higher than girls in computing confidence and, quite interestingly, lower than girls in the statement that "girls can do just as well as boys in computing" (p. 88).

Other than outcome measures, gender difference in the learning processes was also revealed based on observation. Wu (2018) reported that male and female students preferred different types of games in the game design activities: Boys preferred games of sports, shooting, and action, while girls liked role-playing and puzzle games more. Leonard, Buss, Gamboa, et al. (2016) found that girls were more likely to lose interest than boys after working on a single form of game programming with only tutorial but no teacher scaffolding for ten weeks.

Other equity-related issues. Four studies paid attention to equity-related issues other than gender. Feldhausen et al. (2018) investigated the effect of socioeconomic status on students' self-efficacy in CT, and found no significant difference between students from different socioeconomic backgrounds. Duncan and Bell (2015) reported that students from different ethnic groups demonstrated similar levels of interest and engagement in the CT-integrated course, according to teachers' observation.

With a total of 76 middle school participants from rural communities (6 were Native Americans), Leonard, Buss, Gamboa, et al. (2016) reported that students' game products

reflected their cultural tradition (e.g., a Native American student named a game character as Nighthawk) and elements of their living environments (e.g., locally common pickup truck, and local foothills). In addition, they mentioned the relatively high attrition rate of American Indian students compared with that of all participants. In the same context of a larger study, Leonard et al. (2018) presented three focal teachers' case studies. One focal teacher, an African American male, reflected in his teaching journal that he might have made his Native American students uncomfortable by asking them to incorporate their cultural imagery in their game products because it was against their culture; he realized that they avoided more communication about it, a typical Native American reaction when they did not want to further offend the teacher.

Prior experience is typically tied to opportunities and access, and five studies in our collection involved research on prior programming experience. Feldhausen et al. (2018) found that middle school students who had previously attended a STEM event (possibly with more exposure to various programming environments) were more likely to demonstrate higher self-efficacy in CT than those who had not. Similarly, Grover et al. (2015) reported that prior programming experience was positively related with performance in the tests of computational knowledge and programming administered at the end of course. In contrast, Kalogiannakis and Papadakis (2017) compared the survey results of the acceptance of using ScratchJr to learn CT and to incorporate CT in their future science classroom between pre-service teachers who had prior programming experience and those who did not, and there was no significant difference between them.

Although the aforementioned studies reported differences in terms of measured outcomes or observed processes between different student groups, most of them didn't provide or research specific *instructional strategies* on promoting equity. The study by Mouza et al. (2016) was one exception. They set gender role models for middle school

participants: a combination of one female and one male undergraduate student majoring in computer science, was assigned to each group of students as main instructors, despite that there were fewer female undergraduates in computer science.

Discussion

Operationalizing and Assessing CT in STEM Contexts

The findings on defining CT in STEM education reflect some consensus on the broad conceptualization of CT, the trend of redefining CT in STEM contexts, and implications for future research. First, most of the reviewed studies acknowledge the broad conceptualization of CT as a set of skills or practices for problem solving and its essential components including abstraction, decomposition, generalization, algorithm, and so forth.

Second, various definitions or frameworks of CT have been created based on its widely acknowledged broad conceptualization to serve the research and practical demands under different disciplinary contexts. For example, scientific practices play an important role in science learning because they provide learners with the opportunities to experience the process of developing scientific knowledge and improving relevant skills simultaneously. Consistent with this perspective, framing CT as component practices (Weintrop et al., 2016) has become popular in the context of science education. Yet, the operationalization of CT in other STEM disciplines (e.g., How is CT connected with design thinking in engineering education?) requires further research, echoing what Li et al. (2020) proposed that CT should be treated as discipline-specific thinking practice.

Third, besides various STEM disciplines, our finding suggests that researchers and practitioners need to attend to more factors in order to operationalize CT in STEM, including the specific learning environments and characteristics of learners. For instance, for participants at different grade levels, CT should be operationalized differently by considering age appropriateness. One strategy is to employ different programming environments (e.g.,

block-based vs text-based), but considerations should be given to transform CT in different formats according to the features of these environments.

In addition, developing operational definitions of CT is necessary for the purpose of assessment in both research and practice. From the reviewed studies, we observed the vagueness and uncertainty about what and how to assess learning in the integrated CT and STEM educational environment. With a generic definition, researchers and instructors found it difficult to connect CT with STEM and therefore, could not tap into the deep meaning and significance of CT in STEM contexts. As a result, their common solutions were to assess the affective domain (Rubinstein & Chor, 2014) or assess CT and STEM contents separately. Connected with STEM, an operational definition of CT in STEM will provide specific directions when it comes to CT assessment (e.g., using CT to solve specific STEM problems). Our review suggests that more efforts should be devoted to developing assessment tools that target directly integrated CT and STEM learning.

Finally, more solid operational definitions and their accompanying assessments will allow more systematic comparison and contrast of results across studies, moving the field toward a more coordinated direction.

Instructional Strategies for Integrating CT in STEM Education

Our review finds that multiple topic contexts have been used to integrate CT in STEM education and the most popular ones include robotics, game design, and computational modelling. Robotics and game design are more often used in technology education, whereas computational modelling occurs more often in science classrooms. These application contexts also naturally fit the pedagogy of PBI. In contrast, there is a relatively smaller number of studies that use mathematics as contexts to integrate CT. As an important school subject, mathematics provides considerable opportunities and depth for students to develop CT, considering the close connection between computation and mathematics (Wiedemann et al.,

2020). Therefore, we suggest more exploration of integrating CT and math as a future research direction.

While most of the CT activities described in our reviewed studies are computer-based, there have been attempts to incorporate embodied or unplugged activities when integrating CT and STEM instruction. This is especially so for young students (Cateté et al., 2018; Farris et al., 2016; Feldhausen et al., 2018; Mouza et al., 2016; Sung et al., 2017). This line of work attends to the needs of learners at their early stage of CT development, contributing to an evidence-based learning progression of CT. The learning progression of CT has been studied under the contexts of robotic programming (Sullivan & Heffernan, 2016) and Scratch programming (Seiter & Foreman, 2013), as well as by analysing the potential connections among multiple types of CT-integrated activities (Lee & Malyn-Smith, 2020); yet, it is an underdeveloped, albeit important, research area. We call for more empirical research to map out the learning progression for integrating CT in STEM education, paying particular attention to the connections between different phases (e.g., how would unplugged activities contribute to development of CT using a computer-based tool?).

In terms of scaffolding strategy, very few studies among our reviewed articles have investigated the effectiveness of scaffolding through controlled experiments. With limited empirical evidence, the effectiveness of faded scaffolding, one-on-one guidance, and adaptive scaffolding needs further investigation. Besides, researchers may be recognizing the potential of explicitly teaching CT concepts to learners, as about a quarter of the studies included explicit instruction of CT concepts in their curriculum. Yet, no direct research evidence was presented in these studies to suggest that explicit explanation and direct instruction would lead to improved CT learning. We deem explicit instruction as an important aspect of integrating CT in STEM education and call for more research to understand its effect.

The success of any effective instructional innovation depends on teachers' successful implementation in classrooms. Therefore, teachers should be prepared for integrating CT into STEM classrooms. Eleven articles in our review pool focused on pre-service teacher education or in-service teacher professional development of integrating CT in STEM education. The general observation is that the majority of these studies focused on improving teachers' fundamental understanding of CT and/or their affect related to CT, whereas very few studies paid attention to enhancing teachers' CT competency and their ability to integrate CT into their teaching in real classrooms. Borrowing the idea of PCK (Shulman, 1986), we suggest that teacher preparation programs should take one step further to equip pre-/in-service teachers with pedagogical CT-STEM knowledge.

Research for Broadening Participation in Integrated CT and STEM

Our review shows that early attempts have been made to study equity issues in the field of integrating CT in STEM education. Research started to pay attention to the performance or process differences among different student groups, including groups related to gender, race, ethnicity, geographic location, and socioeconomic status. Among these, gender difference has gained the most attention. A common theme among the studies was that while no statistically significant gender difference was found in CT competency, females often had significantly lower confidence or self-efficacy in CT than males.

As one can see, the research on broadening participation in integrating CT in STEM education is far from adequate. One area that needs immediate action is to design and research different instructional strategies, among other means, that promote equitable learning when integrating CT in STEM fields. Some pedagogies have been proposed and investigated, such as setting up role models, grouping students with different levels of prior programming experience. Yet, the effectiveness and efficacy of these strategies need investigation with more carefully designed research.

Limitation

Due to the broad conceptualizations of CT and a wide range of application field, our literature collection may have missed some relevant studies. The reader should keep in mind that our literature search should be taken as a sampling process as opposed to an exhaustive one. Besides, in the process of completing this review, more studies have been published. Therefore, this review, as any review study, has a time limitation.

Conclusion

We reviewed 55 papers with empirical studies on the topic of integrating CT into STEM education. Our major findings from these papers include: (a) attempts have been made to create domain-specific CT definitions in STEM education; (b) problem-based instruction, including application contexts such as game design, robotics, and computational modelling, has been the most popular model in integrating CT in STEM; (c) the assessments of student learning in integrated CT and STEM education targeted different objectives with different formats, but assessment of *integrated* CT and STEM is limited; (d) learning performance and processes between different groups were studied, but pedagogical design and efficacy on enhancing equity was much less researched. Although our review indicates that this area is still in its early stage, our review provides a useful resource to future research and practices, especially in areas such as operationalizing CT in STEM contexts, assessing integrated CT and STEM learning, and research for broadening participation in CT-integrated STEM education.

Compliance with Ethical Standards

Ethical Approval and informed consent

This study does not involve any human participants or animals since we only worked with literature we collected.

Conflict of Interest

The authors declare that they have no conflict of interest.

References

Aho, A. V. (2012). Computation and computational thinking. *The Computer Journal, 55*(7), 832–835.

Aksit, O., & Wiebe, E. N. (2020). Exploring Force and Motion Concepts in Middle Grades Using Computational Modeling: a Classroom Intervention Study. *Journal of Science Education and Technology, 29*(1), 65–82. doi:10.1007/s10956-019-09800-z

Arastoopour Irgens, G., Dabholkar, S., Bain, C., Woods, P., Hall, K., Swanson, H., . . . Wilensky, U. (2020). Modeling and measuring high school students' computational thinking practices in science. *Journal of Science Education and Technology, 29*(1), 137–161. doi:10.1007/s10956-020-09811-1

Bakker, A., Smit, J., & Wegerif, R. (2015). Scaffolding and dialogic teaching in mathematics education: Introduction and review. *ZDM Mathematics Education, 47*(7), 1047–1065.

Barr, V., & Stephenson, C. (2011). Bringing computational thinking to K–12: What is involved and what is the role of the computer science education community? *ACM Inroads, 2*(1), 48–54.

Barth-Cohen, L. A., Montoya, B., & Shen, J. (2019). Making in the middle: Walk like A Robot. *Science Scope,* 42 (9), 12-16.

Basu, S., Biswas, G., & Kinnebrew, J. S. (2017). Learner modeling for adaptive scaffolding in a Computational Thinking-based science learning environment. *User Modeling and User-Adapted Interaction, 27*(1), 5–53. https://doi.org/10.1007/s11257-017-9187-0.

Bean, N., Weese, J., Feldhausen, R., & Bell, R. S. (2015). *Starting from scratch: Developing a pre-service teacher training program in computational thinking.* Paper presented at the 2015 IEEE Frontiers in Education Conference (FIE), EI Paso, Texas, USA.

Benton, L., Hoyles, C., Kalas, I., & Noss, R. (2017). Bridging primary programming and mathematics: Some findings of design research in England. *Digital Experiences in Mathematics Education, 3*(2), 115–138. https://doi.org/10.1007/s40751-017-0028-x.

Berland, M., & Wilensky, U. (2015). Comparing virtual and physical robotics environments for supporting complex systems and computational thinking. *Journal of Science Education and Technology, 24*(5), 628–647. doi:10.1007/s10956-015-9552-x.

Bers, M. U., Flannery, L., Kazakoff, E. R., & Sullivan, A. (2014). Computational thinking and tinkering: Exploration of an early childhood robotics curriculum. *Computers & Education, 72*, 145–157. doi:10.1016/j.compedu.2013.10.020

Bortz, W. W., Gautam, A., Tatar, D., & Lipscomb, K. (2020). Missing in Measurement: Why Identifying Learning in Integrated Domains Is So Hard. *Journal of Science Education and Technology, 29*(1), 121–136. doi:10.1007/s10956-019-09805-8

Boticki, I., Pivalica, D., & Seow, P. (2018). *The use of computational thinking concepts in early primary school.* Paper presented at the International Conference on Computational Thinking Education in 2018, Hong Kong, China.

Bremner, A. (2013). Singing and gaming to math literacy. *Teaching Children Mathematics, 19*(9), 582–584. doi:10.5951/teacchilmath.19.9.0582

Brennan, K., & Resnick, M. (2012). *New frameworks for studying and assessing the development of computational thinking*. Paper presented at the 2012 Annual Meeting of the American Education Researcher Association, Vancouver, Canada.

Çalık, M., & Sözbilir, M. (2014). Parameters of content analysis. *Education and Science, 39*(174), 33–38. http://dx.doi.org/10.15390/EB.2014.3412

Campbell, L. O., & Heller, S. (2019). Building computational thinking: Design and making in teacher education. In J. Leonard, A. C. Burrows, & R. Kitchen (Eds.), *Recruiting, Preparing, and Retaining STEM Teachers for a Global Generation* (pp. 163–189). Leiden, The Netherlands: Brill | Sense.

Cateté, V., Lytle, N., Dong, Y., Boulden, D., Akram, B., Houchins, J., . . . Boyer, K. (2018). *Infusing computational thinking into middle grade science classrooms: Lessons learned*. Proceedings of the 13th Workshop in Primary and Secondary Computing Education, Potsdam, Germany.

Chen, G., Shen, J., Barth-Cohen, L., Jiang, S., Huang, X., Eltoukhy, M. (2017). Assessing elementary students' computational thinking in everyday reasoning and robotics programming. *Computer & Education, 109,* 162-175.

Code.org, CSTA, & ECEP Alliance. (2020). *2020 State of computer science education: Illuminating Disparities*. Retrieved from https://advocacy.code.org/stateofcs

College Board. (2020). AP computer science principles: Course and exam description. Retrieved from https://apcentral.collegeboard.org/courses/ap-computer-science-principles/course

Devolder, A., van Braak, J., & Tondeur, J. (2012). Supporting self-regulated learning in computer-based learning environments: Systematic review of effects of scaffolding in the domain of science education. *Journal of Computer Assisted Learning, 28*(6), 557–573. doi:10.1111/j.1365-2729.2011.00476.x

Djambong, T., & Freiman, V. (2016). *Task-based assessment of students' computational thinking skills developed through visual programming or tangible coding environments.* Paper presented at the 13th Conference on Cognition and Exploratory Learning in Digital Age (CELDA 2016), Mannheim, Germany.

Duncan, C., & Bell, T. (2015). *A pilot computer science and programming course for primary school students*. Paper presented at the Workshop in Primary and Secondary Computing Education, London, United Kingdom.

Falloon, G. (2016). An analysis of young students' thinking when completing basic coding tasks using Scratch Jnr. on the iPad. *Journal of Computer Assisted Learning, 32*(6), 576–593. doi:10.1111/jcal.12155

Farris, A. V., Dickes, A. C., & Sengupta, P. (2016). *Development of disciplined interpretation using computational modeling in the elementary science classroom*. Proceedings of the 12th International Conference of the Learning Sciences, Singapore.

Farris, A. V., & Sengupta, P. (2014). *Perspectival computational thinking for learning physics: A case study of collaborative agent-based modeling*. Proceedings of the 11th International Conference of the Learning Sciences, Boulder, Colorado, USA.

Feldhausen, R., Weese, J. L., & Bean, N. H. (2018). *Increasing student self-efficacy in computational thinking via STEM outreach programs*. Proceedings of the 49th ACM Technical Symposium on Computer Science Education - SIGCSE '18, Baltimore, Maryland, USA.

Gadanidis, G., Cendros, R., Floyd, L., & Namukasa, I. (2017). Computational thinking in mathematics teacher education. *Contemporary Issues in Technology and Teacher Education, 17*(4), 458–477.

Gadanidis, G., Clements, E., & Yiu, C. (2018). Group theory, computational thinking, and young mathematicians. *Mathematical Thinking and Learning, 20*(1), 32–53.

Grover, S., & Pea, R. (2013). Computational thinking in K–12: A review of the state of the field. *Educational Researcher, 42*(1), 38–43. doi:10.3102/0013189X12463051

Grover, S., Pea, R., & Cooper, S. (2015). Designing for deeper learning in a blended computer science course for middle school students. *Computer Science Education, 25*(2), 199–237. Retrieved from https://doi.org/10.1080/08993408.2015.1033142.

Hadad, R., Thomas, K., Kachovska, M., & Yin, Y. (2020). Practicing formative assessment for computational thinking in making environments. *Journal of Science Education and Technology, 29*(1), 162–173. doi:10.1007/s10956-019-09796-6

Hambrusch, S., Hoffmann, C., Korb, J. T., Haugan, M., & Hosking, A. L. (2009). *A multidisciplinary approach towards computational thinking for science majors.* Paper presented at the 40th ACM Technical Symposium on Computer Science Education, Chattanooga, TN, USA.

Harrison, A., Hulse, T., Manzo, D., Micciolo, M., Ottmar, E., & Arroyo, I. (2018, June). *Computational thinking through game creation in STEM classrooms.* Paper presented at the 19th International Conference on Artificial Intelligence in Education, 2018, London, UK.

Hoover, A. K., Barnes, J., Fatehi, B., Moreno-Leon, J., Puttick, G., Tucker-Raymond, E., & Harteveld, C. (2016). *Assessing computational thinking in students' game designs*. Proceedings of the 2016 Annual Symposium on Computer-Human Interaction in Play Companion Extended Abstracts, Austin, Texas, USA.

Hutchins, N. M., Biswas, G., Maróti, M., Lédeczi, Á., Grover, S., Wolf, R., . . . McElhaney, K. (2020). C2STEM: a System for Synergistic Learning of Physics and Computational Thinking. *Journal of Science Education and Technology, 29*(1), 83–100. doi:10.1007/s10956-019-09804-9

ISTE, & CSTA. (2011). Operational definition of computational thinking for K–12 education. Retrieved from https://id.iste.org/docs/ct-documents/computational-thinking-operational-definition-flyer.pdf?sfvrsn=2

Jaipal-Jamani, K., & Angeli, C. (2017). Effect of robotics on elementary preservice teachers' self-efficacy, science learning, and computational thinking. *Journal of Science Education and Technology, 26*(2), 175–192. doi:10.1007/s10956-016-9663-z

Jenson, J., & Droumeva, M. (2016). Exploring media literacy and computational thinking a game maker curriculum study. *Electronic Journal of e-Learning, 14*(2), 111–121.

Kalogiannakis, M., & Papadakis, S. (2017). *A proposal for teaching ScratchJr programming environment in preservice kindergarten teachers*. Paper presented at the 12th Conference of the European Science Education Research Association (ESERA), Dublin, Ireland.

Lamprou, A., & Repenning, A. (2018). *Teaching how to teach computational thinking*. Proceedings of the 23rd Annual ACM Conference on Innovation and Technology in Computer Science Education, Larnaca, Cyprus.

Lee, I., & Malyn-Smith, J. (2020). Computational thinking integration patterns along the framework defining computational thinking from a disciplinary perspective. *Journal of Science Education and Technology, 29*(1), 9-18. doi:10.1007/s10956-019-09802-x

Lee, I., Grover, S., Martin, F., Pillai, S., & Malyn-Smith, J. (2020). Computational thinking from a disciplinary perspective: Integrating computational thinking in K-12 science, technology, engineering, and mathematics education. *Journal of Science Education and Technology, 29*(1), 1-8. doi:10.1007/s10956-019-09803-w

Leonard, J., Buss, A., Gamboa, R., Mitchell, M., Fashola, O. S., Hubert, T., & Almughyirah, S. (2016). Using robotics and game design to enhance children's self-efficacy, STEM attitudes, and computational thinking skills. *Journal of Science Education and Technology, 25*(6), 860–876. doi:10.1007/s10956-016-9628-2

Leonard, J., Buss, A., Unertl, A., & Mitchell, M. (2016). *Using robotics and game design to promote pathways to STEM*. Paper presented at the 38th Annual Meeting of the North American Chapter of the International Group for the Psychology of Mathematics Education, Tucson, AZ: The University of Arizona.

Leonard, J., Mitchell, M., Barnes-Johnson, J., Unertl, A., Outka-Hill, J., Robinson, R., & Hester-Croff, C. (2018). Preparing teachers to engage rural students in computational thinking through robotics, game design, and culturally responsive teaching. *Journal of Teacher Education, 69*(4), 386–407. doi:10.1177/0022487117732317

Lewis, C. M., & Shah, N. (2012). *Building upon and enriching grade four mathematics standards with programming curriculum*. Proceedings of the 43rd ACM Technical Symposium on Computer Science Education, Raleigh, North Carolina, USA.

Li, Y., Schoenfeld, A. H., diSessa, A. A., Graesser, A. C., Benson, L. C., English, L. D., & Duschl, R. A. (2020). On computational thinking and STEM education. *Journal for STEM Education Research, 3*(2), 147–166. doi:10.1007/s41979-020-00044-w

Martin, F. (2018). Rethinking computational thinking. Retrieved October 24, 2020 from http://advocate.csteachers.org/2018/02/17/rethinking-computational-thinking/

Mouza, C., Marzocchi, A., Pan, Y.-C., & Pollock, L. (2016). Development, implementation, and outcomes of an equitable computer science after-school program: Findings from middle-school students. *Journal of Research on Technology in Education, 48*(2), 84–104. doi:10.1080/15391523.2016.1146561

Namdar, B., & Shen, J. (2015). Modeling oriented assessment in K-12 science education: A synthesis of research from 1980 to 2013 and new directions. *International Journal of Science Education,* 37 (7), 993-1023.

National Research Council. (2012). *A framework for K–12 science education: Practices, crosscutting concepts, and core ideas*. Washington, DC: The National Academies Press.

NGSS Lead States. (2013). *Next generation science standards: For states, by states*. Washington, DC: The National Academies Press.

Orton, K., Weintrop, D., Beheshti, E., Horn, M., Jona, K., & Wilensky, U. (2016). *Bringing computational thinking into high school mathematics and science classrooms*. Paper presented at the Transforming Learning, Empowering Learners: The International Conference of the Learning Sciences (ICLS) 2016, Singapore.

Pei, C., Weintrop, D., & Wilensky, U. (2018). Cultivating computational thinking practices and mathematical habits of mind in Lattice Land. *Mathematical Thinking and Learning, 20*(1), 75–89. https://doi.org/10.1080/10986065.2018.1403543

Repenning, A., Webb, D., & Ioannidou, A. (2010). *Scalable game design and the development of a checklist for getting computational thinking into public schools*. Proceedings of the 41st ACM Technical Symposium on Computer Science Education, Milwaukee, Wisconsin, USA.

Román-González, M., Pérez-González, J.-C., & Jiménez-Fernández, C. (2017). Which cognitive abilities underlie computational thinking? Criterion validity of the Computational Thinking Test. *Computers in Human Behavior, 72*, 678–691.

Rubinstein, A., & Chor, B. (2014). Computational thinking in life science education. *PLOS Computational Biology, 10*(11), e1003897. doi:10.1371/journal.pcbi.1003897

Schwarz, C. V., & White, B. Y. (2005). Metamodeling knowledge: developing students' understanding of scientific modeling. *Cognition and Instruction, 23*(2), 165–205.

Seiter, L., & Foreman, B. (2013). *Modeling the learning progressions of computational thinking of primary grade students*. Proceedings of the 9th Annual International ACM Conference on International Computing Education Research, San Diego, San California, USA.

Sengupta, P., Kinnebrew, J. S., Basu, S., Biswas, G., & Clark, D. (2013). Integrating computational thinking with K–12 science education using agent-based computation: A theoretical framework. *Education and Information Technologies, 18*(2), 351–380.

Shen, J., Chen, G., Barth-Cohen, L., Jiang, S., & Eltoukhy, M. (2020) Connecting students' computational thinking in everyday reasoning and programming: Designing a humanoid robotics curriculum for elementary school students. *Journal of Research on Technology in Education.*

Shulman, L.S. (1986). Those who understand: Knowledge growth in teaching. *Educational Researcher*, 15(2), 4–31.

Shute, V. J., Sun, C., & Asbell-Clarke, J. (2017). Demystifying computational thinking. *Educational Research Review, 22*, 142–158. doi:10.1016/j.edurev.2017.09.003

Snyder, H. (2019, November). Literature review as a research methodology: An overview and guidelines. *Journal of Business Research, 104*, 333-339. https://doi.org/10.1016/j.jbusres.2019.07.039

Sullivan, F. R., & Heffernan, J. (2016). Robotic construction kits as computational manipulatives for learning in the STEM disciplines. *Journal of Research on Technology in Education, 48*(2), 105–128. doi:10.1080/15391523.2016.1146563

Sung, W., Ahn, J., & Black, J. B. (2017). Introducing computational thinking to young learners: Practicing computational perspectives through embodiment in mathematics education. *Technology, Knowledge and Learning, 22*(3), 443–463.

Swanson, H., Anton, G., Bain, C., Horn, M., & Wilensky, U. (2019). Introducing and Assessing Computational Thinking in the Secondary Science Classroom. In S.-C. Kong & H. Abelson (Eds.), *Computational Thinking Education* (pp. 99–117): Springer, Singapore.

Syed, M., & Nelson, S. C. (2015). Guidelines for Establishing Reliability When Coding Narrative Data. *Emerging Adulthood, 3*(6), 375–387. https://doi.org/10.1177/2167696815587648

Tang, X., Yin, Y., Lin, Q., Hadad, R., & Zhai, X. (2020). Assessing computational thinking: A systematic review of empirical studies. *Computers & Education, 148*, 103798.

Weintrop, D., Beheshti, E., Horn, M., Orton, K., Jona, K., Trouille, L., & Wilensky, U. (2014). *Defining computational thinking for science, technology, engineering, and math*. Paper presented at the 2014 Annual Meeting of the American Educational Research Association, Philadelphia, Pennsylvania. https://ccl.northwestern.edu/papers/2014/CT-STEM_AERA_2014.pdf

Weintrop, D., Beheshti, E., Horn, M., Orton, K., Jona, K., Trouille, L., & Wilensky, U. (2016). Defining computational thinking for mathematics and science classrooms. *Journal of Science Education and Technology, 25*(1), 127–147.

Wiedemann, K., Chao, J., Galluzzo, B., Simoneau, E., (2020). Mathematical modeling with R: embedding computational thinking into high school math classes. *ACM Inroads, 11* (1), 33-42.

Wilensky, U., & Reisman, K. (2006). Thinking like a wolf, a sheep, or a firefly: Learning biology through constructing and testing computational theories—An embodied modeling approach. *Cognition and Instruction, 24*(2), 171–209.

Wing, J. M. (2006). Computational thinking. *Communications of the ACM, 49*(3), 33–35.

Wing, J. M. (2008). Computational thinking and thinking about computing. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences, 366*(1881), 3717–3725. https://doi.org/10.1098/rsta.2008.0118.

Witherspoon, E. B., Higashi, R. M., Schunn, C. D., Baehr, E. C., & Shoop, R. (2017). Developing computational thinking through a virtual robotics programming curriculum. *ACM Transactions on Computing Education, 18*(1), 1–20.

Wu, M. L. (2018). Educational game design as gateway for operationalizing computational thinking skills among middle school students. *International Education Studies, 11*(4), 15–28. doi:10.5539/ies.v11n4p15

Yin, Y., Hadad, R., Tang, X., & Lin, Q. (2020). Improving and assessing computational thinking in maker activities: The integration with physics and engineering learning. *Journal of Science Education and Technology, 29*(2), 189–214.